Exhibit C

Johnson I text compared to
U.S. Patent No. 5,970,149
(Appl. No. 08/892,982)

REFERENCE TO PENDING APPLICATIONS

[0001] This application is a continuation-in-part of (a) U.S. patent application Ser. No. 09/239,425 entitled "A Secure Electronic Mail System" filed on Jan. 28, 1999 and (b) Ser. No. 09/255,837 entitled "Method For Information Encoding And Transfer" filed on Feb. 23, 1999 which are continuation-in-part applications of co-pending U.S. patent application Ser. No. 08/892,982, filed Jul. 15, 1997, and entitled "Combined Remote Access and Security System"; which is a continuation-in-part of U.S. patent application Ser. No. 08/752,249, filed Nov. 19, 1996, and entitled "Combined Remote Access and Security System".

REFERENCE TO MICROFICHE APPENDIX

[0002] This application is not referenced in any microfiche appendix.

TECHNICAL FIELD OF THE INVENTION

[0003] The invention relates generally to a data processing system, method and article of manufacture allowing for the dynamic reconfiguration of an input/output device controller. In particular, the present invention relates to a computer-based system, method and article of manufacture which supports and facilitates a remote configuration and utilization of an emulated input/output device controller via encrypted data communication between a plurality of users and said controller.

BACKGROUND OF THE INVENTION

~~1. Field of the Invention~~

~~The present invention relates to a system that will provide remote access to allow servicing of a mainframe computer site while at the same time providing for security and integrity of the mainframe computer installation. In particular, the present invention is directed to a system wherein service and maintenance of the mainframe computer system is controlled and monitored from a remote location and service on the mainframe computer system may be performed by a remote support person at a further remote location.~~

~~2. Prior Art~~

~~Current mainframe processing environments use an operator console to display messages about the system. These messages are monitored and any problems are noted. Programmers and other technicians may then become involved in solving a problem. The problem may be beyond the operations staff's ability to handle.~~

~~The mainframe computer system may be serviced and monitored from a remote location. Remote support of mainframe computer installations is becoming increasingly important. This includes both remote monitoring and service support of mainframe computer~~

systems. Businesses have been established which are capable of monitoring and maintaining a wide variety of mainframe computer installations.

From time to time, when problems are found, it is necessary for a technician, field engineer, or remote support person to have access to the mainframe computer system. A technician or field engineer can work on the problems on site at the mainframe installation. With high speed, broad band communications, it is possible for a remote support person or field engineer to diagnose and solve mainframe computer problems from a remote location by communication from a personal computer. Accordingly, the remote support person or field engineer may be at any location. These technicians are increasingly specialized and require wide access to the mainframe computer installation.

Moreover, it is increasingly a trend for employees, including those at mainframe computer installations, to work from their homes on personal computers. In this case, the employees' home computers must be connected to the mainframe computer installations.

At the same time, the computer mainframe installation must retain its security and integrity. In the past, while limited access and "firewalls" have sometimes been employed to maintain security, the field engineer or remote support person needs wide access to the mainframe computer to diagnosis and solve the problems.

Typically, the dispatch control center is located in a secure location. This dispatch control center may be at the same physical premises as the mainframe customer site or may be at a separate location remote from the mainframe. The remote support person, however, is often times at an unsecured location and may operate from a laptop or other unsecured central processing unit machine. Additionally, the mainframe computer business has only limited controls over the field engineer. For example, a disgruntled remote support person or field engineer with wide access to the mainframe computer system could cause considerable problems.

With both the dispatch control center and the support person at remote locations from the mainframe computer center, the channels of communication are important. While secure transmission lines are possible to establish, these are expensive over long distances. Additionally, the support person may be mobile.

The development of personal computers, modems (modulator/demodulator devices) and data connections has allowed the growth of many types of computer networks. The Internet, a somewhat public network of networks, has become an increasingly useful pathway for computer communication. There is, however, a concern about the security and integrity of the Internet pathways.

One solution to security on the Internet has been the encryption of data to be transmitted. One type of encryption uses a single "key" which the sender and recipient must keep secret. Another type of popular encryption uses "public-private keys." The first is a public key made available to anyone. The second is a "secret key" which the user must not allow anyone else to see. The public and private keys work in tandem. If the secret

key is stored on a computer system, it is, however, vulnerable.

The same security issues and concerns may also exist on corporate intranets and private networks.

Accordingly, the present invention is directed to an arrangement where a mainframe or mainframes are secured at a customer site and wired to a personal computer with software for console monitoring. The console monitor is in communication with a secure dispatch control center location. The dispatch control center, upon being alerted of a problem, will contact a support person to diagnose and solve the particular problem. A data encryption key is randomly generated and transmitted from the dispatch control center to both the support person's central processing unit and to the console monitor of the mainframe.

It is a further object and purpose of the present invention to provide a remote access and security system using data encryption keys wherein a data encryption key is never transmitted or sent between the remote support person's central processing unit and the mainframe installation.

SUMMARY OF THE INVENTION

In a combined remote access and security system of the present invention, a single mainframe or multiple mainframes are located at a secure location. The mainframe or mainframes are connected to a console monitor central processing unit through a coax or twinax connection.

The console is used to display status messages about the mainframe computer system including errors or critical situations occurring on the computer system. When specified mainframe system alerts or problems occur a warning or alert will be issued. This alert will be communicated from the console to a dispatch control center central processing unit at a remote, secure location.

A dispatcher will monitor any alarm codes received from the mainframe system. The dispatcher will create a trouble ticket for each incoming alarm, assign a field engineer to the problem and call or otherwise contact the field engineer.

Thereafter, the dispatcher will initiate through the dispatch central processing unit, a unique, randomly generated user identification/password pair which is referenced to the assigned problem number. The user identification/password pair is a data encryption key randomly generated by the dispatch central processing unit. The data encryption key is generated from a mathematical algorithm and will be a randomly generated binary code.

The identification/password encryption key is transmitted in two separate transmissions over two separate paths. The data encryption key is communicated from the dispatcher's central processing unit to the field engineer's central processing unit. Additionally, the dispatch central processing unit will also transmit the data encryption key back to the

~~console central processing unit of the mainframe.~~

~~Once the field engineer has been notified and has received the identification/password pair from the dispatch control center, the field engineer will log on and communicate with the console central processing unit.~~

~~Data communicated from the field engineer's central processing unit to the console central processing unit is encrypted with the identification/password key. The data is subsequently decrypted upon receipt at the console monitor central processing unit. Importantly, the password/identification pair does not travel over the~~[0004] The present invention provides for secured, real-time, configuration and utilization of an emulated input/output device controller. The instant invention advances the art by allowing its practice to be supported via an encrypted communications protocol interfacing with, and relying upon, the teachings, practices and claims disclosed in co-pending U.S. patent applications Ser. No. 09/239,425 and 09/255,837 (hereinafter synonymously referred to as "Secure Agent" or "SA").

[0005] Secure Agent Service Overview

[0006] The following overview is provided to facilitate a comprehensive understanding of the teachings of the instant invention. Secure Agent utilizes a secure login sequence wherein a client connects to a Secure Agent server using a key known to both systems and a client connects and presents the server with user identification (as used herein the term "client" refers synonymously to a remote user establishing, and communicating with the instant invention through Secure Agent allocation and encryption processes as taught in the above noted applications). If recognized, the Secure Agent server initiates a protocol whereby the client's identification is verified and subsequent communication is conducted within a secured (encrypted) construct. For purposes of this overview, the term "server" should be considered a hardware configuration represented as a central processing unit wherein Secure Agent, a Host DLL and driver reside, and are executed. The term "DLL" as used herein refers to a Secure Agent host dynamically linked library (a.k.a. Host DLL). The term "DLL" or "dynamically linked library" is used in a manner consistent with that known to those skilled in the art. Specifically, the term "DLL" refers to a library of executable functions or data that can be used by a Windows application. As such, the instant invention provides for one or more particular functions and program access to such functions by creating a static or dynamic link to the DLL of reference, with "static links" remaining constant during program execution and "dynamic links" created by the program as needed.

[0007] The Secure Agent server presents a variable unit of data, such as the time of day, to the client as a challenge. The client must then encrypt that data and supply it back to the server. If the server is able to decrypt the data using the stored client's key so that the result matches the original unencrypted challenge data, the user is considered authenticated and the connection continue. The key is never passed between the two systems and is therefore never at risk of exposure.

[0008] The initial variable unit of data seeds the transmission of subsequent data so that the traffic for each client server session is unique. Further, each byte of data transmitted is influenced by the values of previously sent data. Therefore, the connection is secure across any communication passageway including public networks such as, but not limited to, the Internet. The distance between the client and server is not of consequence but is typically a remote connection. For accountability purposes, the actions of a client may be recorded (logged) to non-volatile storage at almost any detail level desired.

[0009] The access rights of each client (what the client is able to accomplish during a session) is governed by data stored on the Secure Agent server to which the client is associated. As an example, such rights might encompass the ability to administer and utilize the services of the server system, which would, in turn, include capabilities such as adding new client users, changing a user's rights, transferring new code to the server, using a feature (or service) of the server and more.

[0010] Consequently, Secure Agent allows for the transmission of new code to the server and for that code to be implemented upon demand by a client. Such dynamic, real-time implementation in turn, allows for the behavior of the server to be modified. It is to this behavior modification the instant invention addresses its teachings, and thereby advances the contemporary art.

[0011] As will be readily appreciated by those skilled in the art, though the instant invention utilizes encryption/decryption and code recognition technology associated with Secure Agent, an alternative technology may be employed in support of the instant invention without departing from the disclosure, teachings and claims presented herein.

BRIEF SUMMARY OF THE INVENTION

[0012] The present invention is best viewed as comprised of two server components with one or more client subcomponents or sub-processes disclosed in association thereto. It can be further conceptualized that a distinguishable client component exists for each emulated device type recognized by the invention's server, with an individual client supporting the simultaneous use of a plurality of client-side components. As used throughout the instant invention specification and claims, the term "server" is used synonymously with "emulated device controller", "server central processing unit", "server CPU", and "remotely configurable input/output device controller" and the term "client" is used synonymously with "host user", "client central processing unit", "client CPU" and "remote user".

[0013] The invention's lower-most server component layer is a device driver to communicate directly with one or more hardware components attached to one or more computer systems, such as, but not limited to, mainframe computers (a.k.a. host processors). The driver controls the hardware in a manner prescribed by its design, causing it to interact with the other computer systems to which it is connected as if it were one or more device types (emulation). The driver additionally acts as a conduit to a higher level server component that governs the overall behavior of the emulated devices.

This higher level component primarily supplies the driver with new data to provide through the emulated devices to the other computers to which it is connected and accepts data arriving to the emulated devices carried up by the device driver. Both layers predomoninantly operate on a device by device basis. The higher level server component, in turn, serves as the interface between Secure Agent technology and remotely connected clients allowing for the encrypted transmission of all data external to the server.

[0014] Using the example of an IBM 3215 console, a client would connect to a server and request a list of the 3215 devices which shared membership to the user's security groups. The user would select a device and a logical pathway from the mainframe computer to the client's system would become established. The client would communicate through the server layers with the end result of messages transported from a mainframe through an emulated device to the client for presentation within a window on a computer screen. Conversely, commands to the mainframe may be issued at the client's workstation and are transported through to the emulated device then through it to the mainframe.

[0015] Just as a client might have the ability to administer users (i.e. add/remove), a client might be able to modify the presence and behavior of emulated devices, via Secure Agent administrative functions as taught by the afore noted pending patent applications. Allowable configuration ranges and values are verified and enforced according to rules by the server. The various data elements that may be controlled are listed at the bottom of this section. The server disallows modification of the active configuration (apart from device names and their security groups) and forces such modifications to be made to an inactive configuration. This inactive configuration may be swapped with the active configuration (thus activating it) upon demand. Thus, a new configuration may be prepared prior to a decision made to put it into effect. Additional control functionality includes but is not limited to the following:

[0016] Recycling an adaptor that is connected to an external computer system. This is commonly referred to as a Power On Reset or, more simply, a POR.

[0017] Viewing which users are connected to which devices.

[0018] Disconnecting a client user from a device to which he is connected.

[0019] Activating an inactive configuration.

[0020] Copying the active configuration to the inactive configuration in order to make changes based upon the active configuration.

[0021] Purging the inactive configuration in order to start fresh.

[0022] Consequently it is an object of the instant invention to provide for remote control, operation and use of a server Central Processing Unit (CPU).

[0023] A further object of the instant invention is to provide for a secured logon sequence utilizing encrypted data transmission in accordance with the teachings, disclosure and claims of the above noted pending patent applications.

[0024] Yet another object of the instant invention is to insure that all data transferred external of the emulated input/output device controller is encrypted in accordance with the teachings of the above noted pending patent applications.

[0025] A further object of the instant invention is to provide the ability for an administrator to alter and manage the configuration of emulated mainframe peripheral devices.

[0026] A further object of the instant invention is to allow the selective addition or restriction in the presence of devices to one or more host processors such as, but not limited to, mainframe computers.

[0027] Another object of the instant invention is to provide for a configuration specification which provides the ability to arbitrarily name each emulated device and assign it to one or more security groups of which a user must be a member in order to access that particular device.

[0028] An additional object of the present invention is to provide the capability by which an administrator may add and remove one or more users with respect to emulated input/output device allocation.

[0029] Yet another object of the instant invention is to provide a facility by which an administrator may manage the security groups to which a user belongs, thus controlling the access of devices by users at any level desired down to an individual user level.

[0030] A further object of the instant invention is to provide the ability for a user to access and operate an emulated input/output device.

[0031] Yet another object of the instant invention is to provide the facility by which an administrator may effect/implement new device emulation types.

[0032] Another object of the instant invention is to provide support for multiple device types which may be simultaneously supported and operated.

[0033] Responsive to the foregoing challenges, the Applicant has developed an innovative system, method and article of manufacture to remotely configure and utilize an emulated device controller via an encrypted validation communication protocol.

[0034] It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory only, and are not restrictive of the invention as claimed. The accompanying drawings, which are incorporated herein by reference, and which constitute a part of this specification, illustrate certain embodiments

of the invention and, together with the detailed description, serve to explain the principles of the present invention.

[0035] In this respect, before explaining at least one embodiment of the invention in detail, it is to be understood that the invention is not limited in this application to the details of construction and to the arrangement so the components set forth in the following description or illustrated in the drawings. The invention is capable of other embodiments and of being practiced and carried out in various ways. Also, it is to be understood that the phraseology and terminology employed herein are for the purpose of description and should not be regarded as limiting. As such, those skilled in the art will appreciate that the conception, upon which this disclosure is based, may readily be utilized as a basis for the designing of other structures, methods and systems for carrying out the several purposes of the present invention. It is important, therefore that the claims be regarded as including such equivalent constructions insofar as they do not depart from the spirit and scope of the present invention.

[0036] Additional objects and advantages of the invention are set forth, in part, in the description which follows and, in part, will be apparent to one of ordinary skill in the art from the description and/or from the practice of the invention.

[0037] These together with other objects of the invention, along with the various features of novelty which characterize the invention, are pointed out with particularity in the claims annexed to and forming a part of this disclosure. For a better understanding of the invention, its operating advantages and the specific objects ~~connection between the field engineer and the mainframe site.~~attained by its uses, reference would be had to the accompanying drawings, depictions and descriptive matter in which there is illustrated preferred embodiments and results of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

~~FIG. 1 illustrates a simplified schematic view of a combined remote access and security system as set forth in the present invention.~~

~~FIG. 2 illustrates a schematic view of an alternate embodiment of a combined remote access and security system as set forth in the present invention;~~

~~FIGS. 3A, 3B, 4, 5A, 5B and 6 are flow charts illustrating the sequential steps of the present invention; and~~

~~FIGS. 7 through 13 illustrate sub processes of those in FIGS. 3 through 6.~~

~~DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS~~

~~Referring to the drawings in detail, FIG. 1 illustrates one preferred embodiment of a schematic diagram of a combined remote access and security system 10 of the present invention.~~

At a mainframe computer installation, a single mainframe 12 or multiple mainframes are located at a secure location (illustrated by the box 14). In many industries and businesses, large numbers of transactions are processed on an around-the-clock basis. Because of this demand, multiple mainframe central processing units are utilized within a secure computer complex. Access may be limited by physical measures, such as locked rooms, finger printing, and the like.

The mainframe or mainframes are connected to a console monitor central processing unit 16 which typically includes a keyboard 18 and display 20. The console 16 can be connected with the mainframe or mainframes in various ways, such as, by a coax or twinax connections 22.

The console 16, in the present situation, may employ a Windows NT.TM. operating system or other known operating systems. The operating system will have an application program or programs which is in a client-server format and provides console monitoring and console automation features. The application program watches or monitors the console for certain conditions.

The console 16 is used to display status messages about the mainframe computer system and allows the operations staff to control the operations of the mainframe or mainframes. Types of messages displayed may be about errors or critical situations occurring on the computer system. Examples of problems noted may be a tape drive fault or a fault in a chip on a board.

In today's environment, a single console may be responsible for multiple mainframe computers running multiple computer operating systems.

In specified mainframe system alerts, events or problems, the console will issue a warning or alert. This alert will be communicated from the console 16 through a modem and through a communications path, shown by arrow 30, to a dispatch control center, indicated by box 32. In the present embodiment, the communications path may be across the public Internet network. Each computer or machine will have a distinct Internet protocol address. Other communications paths, such as corporate intranets or private networks, are possible within the teachings of the present invention.

In the present embodiment, the secure dispatch control center 32 is located remote from the mainframe site, although the teachings of the invention apply if the dispatch center is at the same location.

The dispatch control center 32 is ordinarily at a secure location. Thus, access to the computer is limited by physical measures such as locked rooms, fingerprinting and the like. Additionally, access to the dispatch central processing unit 34 may require passwords prior to log on procedures. Typically, the dispatch central processing unit 34 includes a keyboard 36 and a display 38. The dispatch central processing unit 34 will be running a client side version of the application program running on the console monitor

16, as previously discussed.

A dispatcher (not shown) will monitor incoming alarm codes received from the mainframe 12. If an alert occurs, it will appear on the display screen 38 of the dispatcher. Upon receipt of an alarm code, it will display in a list on the display screen 38.

The dispatcher will create a trouble ticket for each incoming alarm in the problem tracking program. Alternatively, the procedure to create a problem or trouble ticket might be automated.

Once this has been completed, a field engineer or other remote support person will be assigned to the problem and will be called or otherwise contacted. In one such procedure, the dispatcher will call the field engineer or remote support person via telephone over a voice line. This connection is shown by arrow 40. The field engineer will be assigned a problem number for the incoming problem on the mainframe computer.

Thereafter, the dispatch control center will initiate a utility software program on the dispatch central processing unit 34 which will create a unique, randomly generated user identification/password pair which is referenced to the assigned problem number. In the FIG. 1 embodiments, the user identification/password pair is a data encryption key randomly generated by the dispatch central processing unit 34.

In the present case, the data encryption key is generated from a mathematical algorithm and will be a randomly generated binary code of 128 bits. The data encryption key is also time limited so that after a certain period of time, it will automatically expire. For example, the data encryption key may be valid for a period of 24 hours, after which it is no longer valid.

The identification/password pair is transmitted in two separate transmissions in two separate paths. The data encryption key is communicated and transmitted from the dispatch central processing unit to a remote support person or field engineer central processing unit 50 as shown by arrow 52. The field engineer central processing unit may take many forms, such as a laptop terminal, hand held PC or a desktop computer.

The dispatch central processing unit will also transmit the identification/password data encryption key back to the console central processing unit 16 as shown by arrow 24. The data encryption key is itself also encrypted. The data encryption key is itself decrypted at the field engineer's central processing unit and at the console.

Once the field engineer or remote support person has been notified and has received the identification/password pair from the dispatch control center, the field engineer 50 will log on and communicate with the console central processing unit 16 as shown at arrow 54. The field engineer will be running a client side version of the same application program.

The communication between the field engineer and the console may be made through a

public network such as the Internet. The encrypted data is decrypted at the console monitor.

The field engineer or remote support person will input and download the assigned problem number already received from the dispatch control center 32. The field engineer will thereby retrieve the problem details from the console. The field engineer will, thus, be connected to the mainframe site. Importantly, the password does not travel over the connection between the field engineer central processing unit 50 and the mainframe site 14.

Once connected to the mainframe computer site, the field engineer or remote support person retrieves necessary information through the console central processing unit 16 via the coax 22 connection to the mainframe 12. The field engineer, thus, has access to the mainframe and will endeavor to solve the problem presented.

Once the problem is resolved, the field engineer will notify the dispatch control center 32 that the problem has been resolved as shown at arrow 26. This may be done in a number of ways. This may be done by telephone through voice line. Alternatively, the field engineer may communicate through the field engineer's central processing unit 50 through a communications line back to the dispatch central processing unit. This may also be performed through the Internet.

The dispatcher closes the problem in the problem tracking system. Thereafter, the unique identification/password pair is invalidated so that there is no longer access to the mainframe computer. The dispatcher closes the problem in the dispatch central processing unit database, which then removes the identification/password pair from the console monitor 16 at the mainframe site.

Each of the computer communications may be made through a public network such as the Internet. The data connection from an unsecured terminal/location is at all times secured by the present invention.

FIG. 2 illustrates an alternate embodiment 60 wherein the Internet protocol address are provided dynamically from a secure name server central processing unit.

At a mainframe computer installation, a single mainframe 62 or multiple mainframes will be located at a secure location (illustrated by box 64). The mainframe or mainframes are connected to a console monitor central processing unit 66 which typically includes a keyboard 68 and a display 70. The console 66 can be connected with the mainframe or mainframes in various ways, such as by coax or twinax connections 72.

Alerts, events or problems will be noted by the console which will issue a warning or alert. This alert will be communicated from the console 66 through a modem and through a communications path, shown by arrow 74, to a dispatch control center 76. The dispatch control center includes a dispatch central processing unit 78 having a keyboard 80 and a display 82. The dispatch central processing unit 78 will be running a client side version of

the application program running on the console monitor. If an alert occurs at the console monitor, it will be transmitted and appear on the screen of the dispatch central processing unit. Upon receipt of an alarm code, it will display in a list on the display screen 82. The dispatcher will create a trouble ticket for each incoming alarm in the problem tracking program. Alternatively, the procedure to create a problem or trouble ticket might be automated.

A field engineer or other remote support person will be assigned to the problem and will be called by a telephone or otherwise contacted which is shown by arrow 84. Thereafter, the dispatch control central processing unit 72 will communicate with a secure name server 86 or 88. The secure name server may be located on the premises of the dispatch control center or may be remote therefrom. The secure name server will, through a utility software program, generate a unique, randomly generated user identification/password pair. This will be referenced to the assigned problem number. The user identification/password pair is a data encryption key randomly generated. The data encryption key is transmitted in two separate transmissions over two separate paths. The data encryption key is communicated and transmitted from the dispatch central processing unit 78 to a remote support person or field engineer central processing unit 90 as shown by arrow 92.

The dispatch central processing unit will also transmit the data encryption key back to the console central processing unit 68 which is shown by arrow 94.

After the field engineer or support person has been notified and has received the identification/password pair from the dispatch control center, the field engineer will log on and communicate with the console processing unit 68 as shown by arrow 96.

Once the problem has been resolved, the field engineer or support person will notify the dispatch control center that the problem has been resolved. This is illustrated by arrow 98. The dispatcher at the dispatch control center closes the problem in the problem tracking system. Thereafter, the unique identification/password pair is invalidated so that there is no longer access to the mainframe computer site 64. The dispatcher closes the problem in the dispatch central processing unit database which then removes the identification/password pair from the console monitor 68 at the mainframe site.

FIGS. 3 through 13 illustrate the process of the present invention that will provide remote access to allow servicing of the mainframe computer while providing for security and integrity of the mainframe computer installation. The process will be described in relation to the FIG. 2 embodiment with a pair of dispatch control centers. FIGS. 3A and 3B illustrate the initial process at the secure customer mainframe site 14 to monitor for alerts. After the process has been started, as shown at 100, the console will be checked for alert situations illustrated at box 102.

If there is no unreported alert, as at 104, a check will be made to see whether the reporting interval has expired 106. If the reporting period has expired 106, then the current Internet protocol address (IP) will be registered with a first secure name server, as

seen at 108. If the first secure name server does not register the Internet protocol address, then the current Internet protocol address will be registered with the second secure name server as seen at 110.

Returning to box 104, if there is an unreported alert, an Internet protocol address will be obtained for the first dispatch control center from a secure name server central processing unit as shown at 112. The secure name server is a repository of customer sites and their current IP addresses. Once the Internet protocol address has been obtained for dispatch center 1, an alert will be reported to the first dispatch center, as seen at 116.

If the report on the alert has been received, box 118, then the process can continue. If there is no success, then, as shown on FIG. 3B, an Internet protocol address will be obtained for dispatch center 2 from either secure name server, as shown at 120. If an internal protocol address has been obtained for the second dispatch center as shown at 122, the alert will be reported to the second dispatch center as shown at box 124. If the alert is reported as shown at 126, the process will again continue in same manner.

FIG. 4 illustrates the process for a dispatch control center to handle an incoming alert from a secure mainframe customer site. The FIG. 4 process would chronologically follow the process described in FIGS. 3A and 3B. The dispatch control center will receive an alert from the mainframe customer site 130. A problem ticket or problem number will be created in a tracking system as shown at box 132. A unique user ID/password pair for the remote support person will be generated, as at box 134. An Internet protocol address for the customer site will be obtained from a secure name server, as seen in box 136. Obtaining an IP address for the customer site will be explained in detail below.

Once the Internet protocol address has been obtained for the customer site as shown at 138, a connection will be made from the dispatch control center to the customer site as shown at 140.

The remote support person's user ID/password pair will be set up on the customer mainframe site 142. After the connection with the customer site has been disconnected 144, a remote support person will be selected from an availability list 146. The remote support person may be contacted in various fashions, such as by telephone, and given the problem number 148.

FIG. 5 illustrates the process for the remote support person or field engineer that would be employed to handle the problem that has been reported. This procedure would chronologically follow the process shown in FIG. 4.

As seen in FIG. 5A, once a problem has been received from the dispatch control center as shown at box 152, an Internet protocol address for the dispatch control center will be obtained from one of the secure name servers 154. This process will be explained in detail below.

Once the Internet protocol address has been obtained for the dispatch control center as

shown at 156, a connection will be made to the dispatch center 158. The name and details for the secure mainframe customer site will be provided 160. Thereafter, the remote support person will disconnect from the dispatch control center 162.

An Internet protocol address will be obtained for the customer site from either of the secure name servers 164. Once the Internet protocol address for the mainframe customer site has been obtained 166, the remote support person will connect to the customer site using the Internet protocol as shown at 170. The remote support person or field engineer will be able to work to solve the particular problem as seen at box 172 and, thereafter, disconnect from the mainframe customer site 174.

An Internet protocol address will be obtained for the dispatch control center from either secure name server as shown at 176.

Once the Internet protocol address has been obtained by the support person for the dispatch center 178, the remote support person will connect to the dispatch control center using that Internet protocol address 180. The support person will be able to report completion of the assignment and closing of the problem record 182. The support person will thereafter disconnect from the dispatch center, as shown at 184.

FIG. 6 illustrates the next sequential process in the overall system of the present invention. The dispatch control center will invalidate the remote support person's user ID/password at the secure mainframe customer site.

The dispatch control center will obtain an Internet protocol address for the mainframe customer site from either secure name server, as shown at 190. Once an Internet protocol address has been obtained 192, a connection will be made between the dispatch control center to the console monitor at the customer site using the Internet protocol address as shown at 194. If no Internet protocol address has been obtained, an error will be reported as shown at box 188.

The dispatch center's unique user ID/password will be provided to the console at the customer site, as seen at box 196. A session will thereby be established to the console monitor at the mainframe customer site (198). The remote support person's user ID/password on the customer site console will be invalidated as shown at step 200, following which the session will be disconnected 202.

The remaining processes illustrated in FIGS. 7 through 13 are sub-processes of the foregoing.

FIG. 7 illustrates the process to register a computer with a secure name server central processing unit. A target secure name server will be selected by its Internet protocol address, as shown at box 210. The secure name server will be provided an access user ID/password pair as seen at box 212. A session will thereby be established to the server as shown at 214. If the session has been established 216, the Internet protocol address for the named machine will be registered 218. This process is also seen in FIG. 3A at boxes

~~108 and 110.~~

~~FIG. 8 illustrates the process to obtain an Internet protocol address from a secure name server. This process is shown at box 112 in FIG. 3A. As seen in FIG. 8, a secure name server will be selected by its Internet protocol address, as seen at box 220. The secure name server will be provided with an access user ID/password 222 in order to establish a session 224. Once a session has been established, as shown at 226, an Internet protocol address will be requested for the console monitor 228. If the named computer has been defined 230, a check will be made whether the named machine has its address registered 232, and if the registration is up-to-date 234.~~

~~FIG. 9 illustrates the process for either of two secure name servers to obtain an IP address initially from one server and, if not successful, from a second server. This process would be utilized at 176 in FIG. 5B.~~

~~FIG. 10 illustrates a process to obtain Internet protocol address for a mainframe customer site from initially a first server and, thereafter, a second server for the customer mainframe site.~~

~~FIG. 11 illustrates the subprocess to report an alert from the mainframe customer site to a dispatch center. This step is illustrated in FIG. 3A at box 116.~~

~~The subprocess to connect a remote support person or field engineer to a dispatch center is illustrated in FIG. 12.~~

~~Finally, the subprocess to connect to the console at a mainframe customer site using the Internet protocol address is illustrated in FIG. 13.~~

~~Whereas, the present invention has been described in relation to the drawings attached hereto, it should be understood that other and further modifications, apart from those shown or suggested herein, may be made within the spirit and scope of this invention.~~[0038] FIG. 1 is a system schematic providing a conceptual overview of primary hardware and software components of the instant invention as practiced in its preferred embodiment.

[0039] FIG. 2 is a logic flow diagram illustrating processing steps associated with the server initialization processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0040] FIG. 3 is a logic flow diagram illustrating processing steps associated with the server termination processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0041] FIG. 4 is a logic flow diagram illustrating processing steps associated with the adaptor configuration load processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0042] FIG. 5 is a logic flow diagram illustrating processing steps associated with the client connection processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0043] FIG. 6 is a logic flow diagram illustrating processing steps associated with the client disconnection processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0044] FIG. 7 is a logic flow diagram illustrating processing steps associated with administrative functions given illustrative user response/input strings.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0045] While the making and using of various embodiments of the present invention are discussed in detail below, it should be appreciated that the present invention provides for inventive concepts capable of being embodied in a variety of specific contexts. The specific embodiments discussed herein are merely illustrative of specific manners in which to make and use the invention and are not to be interpreted as limiting the scope of the instant invention.

[0046] While the invention has been described with a certain degree of particularity, it is clear that many changes may be made in the details of construction and the arrangement of components without departing from the spirit and scope of this disclosure. It is understood that the invention is not limited to the embodiments set forth herein for purposes of exemplification, but is to be limited only by the scope of the attached claim or claims, including the full range of equivalency to which each element thereof is entitled. Turning now to FIG. 1.

[0047] In FIG. 1, a server CPU 103 has executing under control of its control program, Secure Agent software 106. The present invention advances the art and improves upon technology taught and claimed in the above noted pending applications, said applications and teachings incorporated by reference herein. The server 103 also has operating under control of its control program the remote configuration software 109 of the instant invention. Embodied within the server 103 is a hardware adaptor card 112. Said adaptor card 112 is in turn communicably attached to one or more host processors (121, 124, 127, 128). As used herein, the term "adaptor" refers synonymously to those hardware configurations such as, but not limited to, "adaptor cards" which allow for connectability between two or more central processing units and the transference of data associated therewith. Illustrative non-limiting examples of such adaptors as used herein would include Crossroads ESCON adaptors, Crossroads ESCON parallel adaptors, Bus-Tech adaptors and IBM ESCON adaptors.

[0048] In FIG. 1, the host processors (121, 124, 127, 128) are illustrated as Host 1 128 executing as its control program a VM system, Host 2 121 operating under as its control program a CICS system, Host 3 124 operating under the controller of its control program

an IMS system and Host 4 127 operating under the dispatching control of its control program (ACP) a plurality of application specific programs. In turn, each of the host processors 128, 121, 124 and 127 illustrated in FIG. 1, have connected to it one or more physical input/output devices 131. In FIG. 1, said input/output devices are depicted as tape drives 141, direct access storage device 138 and smart terminals/personal computer/client computing capabilities 135. Also shown in FIG. 1 is a plurality of clients referred to as Host users 145 which are communicably attached to the server 103 of the instant invention via a communications network 148 such as, but not limited to, the Internet or other computer compatible network wherein computer recognized and generated signals may be communicated between one or more central processing units.

[0049] Lastly shown in FIG. 1 is a Security Administrator client 151 interactively communicating with the Secure Agent software 106 operating within the server 103. As will be discussed in further detail and in association with FIGS. 2 through 7, the Security Administrator 151 utilizes Secure Agent software 106 to administer and maintain user/resource profiles 157 and further communicates with information conveyed to said Secure Agent software 106 via the software processes associated with the remote configuration software 109 of the instant invention.

[0050] For purposes of clarity and to assist in comprehension of the instant invention, it is convenient to view the invention as being comprised of a number of processing subcomponents. Such processing subcomponents include, but are not limited to, Server Initialization Server Termination, Adaptor Configuration Load, Client Communication, Client Termination and Administration related subcomponents.

[0051] The following discussion in association with FIG. 1 provides a brief non-limiting synopsis of the teachings of the instant invention and generally discusses the interrelationships of hardware and software processing components of the instant invention. In FIG. 1, a Security Administrator 151 defines via Secure Agent software 106, user and resource profiles 157. Such profiles are stored in a non-volatile storage medium, such as but not limited to, a disk drive 158. User resource records are those records which typically define security group or groups, and access control variables associated with the user. Stated succinctly, the user resource record/profile defines those resources that the user may utilize and the bounds of such utilization. The Security Administrator 151 may also define resource profiles, such resource profiles define the device type and grouping of emulated input/output devices as well as central processing unit designations associated with each emulated device type and/or grouping. When attempting to establish a session between a host user 145 and any one of the operating systems and/or application programs operating under the dispatching control of the operating systems of host processors 128, 121, 124 or 127, a user via a communications network 148 communicates first with Secure Agent software 106 operating within the server 103 of the instant invention 109. Assuming the user 145 is recognized as an authenticated and authorized user of the system as governed by Secure Agent software 106, the user 145 next requests a device or a device grouping of emulated input/output devices he or she anticipates utilizing in the requested session. The Secure Agent software 106 verifies the user 145 as authority to allocate such emulated input/output

devices and correspondingly associates such devices with the user and user session between one or more of the host processors 128, 121, 124 and 127. Once established, the session continues as normal with input/output requests of the user serviced via emulated input/output device as opposed to the real input/output devices 131 associated with one or more of the host processors. Upon completion of the session or a specific deallocation request initiated by the user, the client termination subprocess of the instant invention deallocates the emulated input/output device or devices. As indicated, the processing subcomponents of the instant invention further include Adaptor Configuration Load, Client Communication, Client Termination, Administration, Server Initialization and Server Termination subprocesses. It is to such subprocesses FIGS. 2 through 7 address themselves. A more detailed disclosure of each subprocess follows.

[0052] FIG. 2 discloses in further detail the process steps in which the server of the instant invention is initialized. While discussion of the individual subprocesses is provided in an illustrative logic sequence, it is to be noted that process steps defined therein need not occur in a serial manner. Rather it is expressly recognized that many of the subprocesses execution steps may be executed in a concurrent manner, or have their execution sequence factored upon the statusing of a previously executed process step.

[0053] Server Initialization, FIG. 2

[0054] With respect to server initialization, the driver of the instant invention first initializes all driver module-wide variables, such as clearing out how many adaptors are being supported, 201. Once these variables have been initialized, adaptors are located by enumerating all peripheral component interconnect computer Bus-type (PCI) devices present in the system using data and techniques published by the PCI Special Interest Group and by Microsoft's Window's NT Device Driver Kit (DDK). Specifically, the adaptor vendor and device IDs 202 are referenced to identify the presence of such supported adaptors. For each adaptor located, adaptor specific variables are initialized by the driver 203, with the resources used by the adaptor, such as buffer areas and IRQ (interrupt request lines) being next allocated and reserved 204 using functions provided by DDK. The adaptor is then reset 205 by the driver using a technique made known by the adaptor's manufacturer. Since these adaptors are generally intelligent it is necessary to transfer (download) to them microcode (a manufacturer-supplied program specific to such a device) that controls internal instruction sequencing. Therefore, microcode is downloaded into the adaptor 206 in a manner prescribed by the adaptor manufacturer with the adaptor then considered initialized 207. The driver next requests a connection to each unique IRQ so that any interrupts generated by any of the recognized adaptors may be serviced by the driver 208 and next initiates timer support 209 so that approximately once every second, general operations may be performed on behalf of each adaptor. This support typically, though not limitedly, includes ensuring an adaptor does not generate a non-detected interrupt. Having once initiated its timer 209, the driver next exposes standard module-wide support to all applications 210, which allows for communications with the driver as to be established by the Host DLL.

[0055] Subsequent to the driver initialization, the Host DLL initializes variables it utilizes

211 and clears a user connection block to allow information for each user to be represented 212. The Host DLL further exposes and makes available to Secure Agent a block of data, representing an emulated device specific administrative instruction set 213, for each user. In addition to such normal data elements as a user ID and password, this instruction set advises Secure Agent to maintain device type and security group strings on behalf of each user specifically for the support of this Host DLL. The device types limits those types of emulated devices to which a user might claim access whereas the device security groups name the emulated device security groups to which a user is subscribed. In addition, at this stage linkage to configuration support routines within the Host DLL is also established. As practiced in one embodiment of the invention, the root name of the administrative tree structure is exposed to Secure Agent indicating that the Host DLL supports the configuration of information and will respond in a positive manner to requests for information and management of branches under this particular root. The Host DLL next creates a mutex serialization mechanism to be used by configuration support routines during access of adaptor configuration data to insure data integrity 214. This serialization mechanism is used to prevent for example potential simultaneous updates by multiple administrators as well as to prevent a client from enumerating emulated devices while it is being manipulated.

[0056] The Host DLL continues to open or otherwise establishes communication with the driver 215 and requests from it a number of recognized adaptors 216 to which the driver responds 217, whereupon the Host DLL requests from the driver its version number 218 to which the driver also responds 219. The Host DLL then records into a Secure Agent log the driver version and the number of adaptors it controls 220, and proceeds to indicate that each adaptor is not yet in a condition to support client connectivity 221. Data representing the adaptor configuration to be utilized (the active configuration) is next loaded 223. This data specifies device types and number of devices to be emulated, in conjunction with user-friendly (readable) names and security groups for each such emulated device. A second unique set of this data is loaded (the inactive configuration) 224 on behalf of this same adaptor to be used as a work area for administrators. This allows administrators to accumulate a series of configuration changes prior to effecting the activation of those changes as a whole. During said initialization, the Host DLL lastly ensures that the loaded adaptor configurations are within operationally permissible parameters 225.

[0057] FIG. 3 is a logic flow diagram illustrating processing steps associated with the server termination processing subcomponents of the instant invention as practiced in its preferred embodiment. Turning now to FIG. 3.

[0058] In FIG. 3 with respect to server termination, the Host DLL first disconnects each currently connected user 301. Such disconnection is facilitated via processing accommodated in the Client Disconnection Processing subcomponent as will be discussed in association with FIG. 6. Recognized adaptors are then set offline to their channels through the Adaptor Configuration Load processing subcomponent 302. The Host DLL next ceases communication, or closes the driver 303, and frees all allocated storage and resources 304. The one second timer is then closed by the driver 305 and

module-wide exposure of support to application through NT is eliminated 306. The driver then ensures/verifies each adaptor is offline to the channel and the adaptor is reset 307, disconnects all previously connected IRQ's 308, and destroys each object instance 309. Such destruction further includes but is not limited to elimination of exposure of the emulated devices support to applications through NT 310 and the freeing of all allocated storage and resources 311.

[0059] FIG. 4 is a logic flow diagram illustrating the processing steps associated with the Adaptor Configuration Load processing subcomponent of the instant invention as practiced in its preferred embodiment.

[0060] In FIG. 4 the Host DLL first indicates the adaptor's unavailability 401 and for each client currently connected to a logical unit on this adaptor, issues a message to the client indicating that the client is being disconnected due to administrative device management 402. The Host DLL then performs the client disconnection services in association with the invention's Client disconnection subprocess as will be discussed in further detail in association with FIG. 6. The Host DLL continues by next recording into Secure Agent log the configuration for this adaptor is being loaded 403 and if the adaptor is to be forced offline to the mainframe to which it is connected 404, prepare and uses an empty configuration indicating that Emulated devices are not to be emulated during this session. If the adaptor is not to be forced offline, an active configuration for the adaptor is provided and a request that the adaptor using the active configuration data is initiated 405. The driver as instructed causes the adaptor to be offline to the channel at this stage in the adaptor configuration load 406, destroys each of the adaptor emulated devices driver object instances 407 causing or eliminating the exposure of emulated devices support to applications through NT 408 and frees all allocated storage and resources 409. The driver next determines if Emulated devices are to be emulated 410 and then request that the adaptor be brought online to the channel 411, lastly indicating that the adaptor is available for client use 412.

[0061] FIG. 5 is a logic flow diagram illustrating the processing steps associated with the Client Connection processing subcomponent of the instant invention as practiced in its preferred embodiment.

[0062] Client Connection, FIG. 5

[0063] In FIG. 5, a client connection first initializes variables that it utilizes 501 then employs Secure Agent client code in order to establish a connection to the Host DLL 502, whereupon the Host DLL retains the client's name 503 and loads the client's device type and security groups 504. A new client object instance is then created to represent this new client connection with the variables it will use becoming initialized 505. The Host DLL then stores the location of the client object in a user connection block 506. At this point the client sends to the Host DLL the command version level that represents the client feature set as a means to facilitate backward compatibility by future Host DLLs 507 which the Host DLL stores for possible reference 508. By knowing the version of the client, the Host DLL can and will prevent communicating with older clients in a manner

supported only by newer clients, whereas newer clients will be able to take advantage of a fuller set of features that the Host DLL offers. The client next provides to the Host DLL the emulated device type in which it is interested 509 whereupon the Host DLL stores it for later reference 510. The client then requests of the Host DLL its command version level 511 that the client stores for possible reference 513. Just as with the Host DLL being able to restrict its behavior for older clients, since the client knows the version level of the Host DLL it can restrict itself from attempting to take advantage of features available only on newer servers whereas newer servers might be more fully exploited. The client then requests from the Host DLL a list of the currently available emulated devices to which the client may connect 514. The Host DLL returns the response back 515 whereupon the client selects one of the emulated devices and requests that the Host DLL establish a connection to it on its behalf 516.

[0064] FIG. 6 is a logic flow diagram illustrating the processing steps associated with Client Disconnection processing subcomponent of the instant invention as practiced in its preferred embodiment.

[0065] As can be seen in FIG. 6, the Host DLL destroys the client object instance which requires the following activity. If connected to a logical unit, the logical unit is closed 601 and the threads that were created to perform input/output device of the logical unit, if any, are terminated 602. If connected to a logical unit, the logical unit-in-use flag is set to not in use 603 and if connected to a logical unit, the logical unit client value is set to none 604. The Host DLL lastly frees all allocated storage and resources for the client object 605.

[0066] Administrative Configuration

[0067] When an administrator desires to modify the configuration of adaptors managed by the Host DLL it issues requests for enumeration of the "/Adaptors" root and its branches to which the Host DLL will respond. This provides the administrator with the means necessary to discover what information exists to be changed. The data exposed through these branches correlates to the data within the active and inactive configurations for each adaptor.

[0068] Once supplied with the name and value of a piece of adaptor configuration data an administrator can decide whether or not to make changes to it and, if so, supply that name with a new value back to the Host DLL which will then make that change on the administrator's behalf.

[0069] Additionally, an administrator may enumerate a series of controls that can be employed for special actions by the Host DLL against an adaptor. Specifically, an administrator might decide to activate the inactive configuration, whereupon the Host DLL will exchange the data of the active configuration with that of the inactive configuration then perform the actions detailed with Adaptor Configuration Load, FIG. 4. If, on the other hand, an administrator opted to copy the contents of the active configuration into that of the inactive configuration then the Host DLL would perform

that action. An administrator also has the option to simply clear out the inactive configuration whereupon the Host DLL would reinitialize it to reflect the absence of configured emulated devices. If an administrator decided it was necessary to reinitialize the adaptor then he could specify that the Host DLL do so whereupon it would perform the actions detailed with Adaptor Configuration Load, FIG. 4. Finally, if an administrator decided that an adaptor should either be kept offline or could come back online then he could request that of the Host DLL and it would toggle that state for the adaptor then perform the actions detailed with Adaptor Configuration Load, FIG. 4.

[0070] Non-limiting examples of dialog and processing as provided for in the invention's administrative configuration subcomponent follow immediately for purposes of facilitating full and enabling disclosure.

[0071] Connected Client Traffic from Logical Unit: Mainframe Message (3215 Example)

[0072] When the adaptor interrupts with a message from the mainframe then that message is first caught by the driver emulated devices object and carried up into the Host DLL by a thread created on behalf of the client that performs I/O against the Logical unit. This message is then transmitted through SA to the client.

[0073] Connected Client Traffic from Logical Unit: Online or Offline Event (3215 Example)

[0074] When the adaptor is found to go online or offline to the channel then that event is first caught by the driver emulated devices object and carried up into the Host DLL by a thread created on behalf of the client that performs I/O against the Logical unit. This event is then transmitted through SA to the client.

[0075] Connected Client Traffic from Client: Mainframe Command (3215 Example)

[0076] The client may send a mainframe command to the Host DLL which is immediately transported to the driver emulated devices object by a thread created on behalf of the client that performs I/O against the Logical unit. The driver emulated devices object then requests that the adaptor send the command to the mainframe.

[0077] FIG. 7 is a logic flow diagram illustrating processing steps associated with administrative functions given non-limiting examples of user input command strings. Turning now to FIG. 7.

[0078] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors

[0079] Administrator requests an enumeration of "/ESCON Adaptors" 701.

[0080] Host DLL builds and returns a string consisting of a concatenation of all the adaptors, in the form of Adaptor # where # is the 1-based number of the adaptor, along

with a flag for each indicating that each element has, in turn, more branches 702.

[0081] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #

[0082] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1" 703.

[0083] Host DLL builds and returns a string consisting of a concatenation of "Active Configuration" and "Inactive Configuration", each with a flag for each indicating that they have, in turn, more branches, along with a string of "Configuration Control" with a flag indicating that it has values 702.

[0084] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/(In)Active Configuration

[0085] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Active Configuration" 704.

[0086] Host DLL builds and returns a string consisting of a concatenation of 16 CUs, in the form of Control Unit x## where ## is hexadecimal from 00 through 0F, along with a flag for each indicating that each element has, in turn, more branches 702.

[0087] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##

[0088] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Active Configuration/Control Unit x00" 705.

[0089] Host DLL builds and returns a string consisting of a concatenation of "Assignments" and "Logical Units", each with a flag indicating that they have values 702.

[0090] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Assignments

[0091] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Inactive Configuration/Control Unit x00/Assignments" 706.

[0092] Host DLL builds and returns a string consisting of a concatenation of the following: 702

[0093] A. "Controller Type" with a flag indicating the data presentation to be a drop-down box.

[0094] This includes a list of all of the valid CUTypes (i.e. 7412, 3174) along with the currently assigned value. This value is taken from the specified Adaptor configuration

data for this adaptor, indexed to the specified control unit.

[0095] B. "Base Address" with a flag indicating the data presentation to be a text box. This includes the currently assigned value. This value is taken from the specified Adaptor configuration data for this adaptor, indexed to the specified control unit.

[0096] C. "Device Count" with a flag indicating the data presentation to be a text box. This includes the currently assigned value. This value is taken from the specified Adaptor configuration data for this adaptor, indexed to the specified control unit.

[0097] D. If the specified Adaptor configuration is the active configuration then a flag is added to all fields marking them as non-modifiable meaning that this data cannot be changed. For these particular datas only that within the inactive configuration may be worked upon.

[0098] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Logical Units

[0099] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Inactive Configuration/Control Unit x00/Logical Units" 707.

[0100] Host DLL builds and returns a string consisting of a concatenation of the following: 702

[0101] A. For each emulated devices per Logical Unit Count for the specified Adaptor configuration data for this adaptor, indexed to the specified control unit (the following uses of ## is the current Logical Unit Count entry+the Logical Unit Base, providing the emulated devices address as it appears to the mainframe.):

[0102] 1. "Device x## Name(s)" with a flag indicating this is a text box. This includes the currently assigned value per the specified Adaptor configuration data for this adaptor, indexed to the specified CU and emulated devices per the current Logical Unit Count entry.

[0103] 2. "Device x## Group(s)" with a flag indicating this is a text box. This includes the currently assigned value per the specified Adaptor configuration data for this adaptor, indexed to the specified CU and emulated devices per the current Logical Unit Count entry.

[0104] 3. If the specified Adaptor configuration is the active configuration:

[0105] a. "Device x## Status" with a flag indicating this is a text box. This includes either the currently assigned emulated devices Client value (client userid) if the emulated devices In-Use flag indicates "in use", otherwise "this device is not in use". The emulated devices values involved are per the specified Adaptor configuration data for this adaptor, indexed to the specified CU and emulated devices per the current Logical Unit Count

entry. This field is marked as non-modifiable meaning that this data cannot be changed (informational only)

[0106] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/Configuration Control

[0107] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Configuration Control" 708.

[0108] Host DLL builds and returns a string consisting of a concatenation of the following: 702

[0109] A. "Check this then click save to activate the inactive config" with a flag indicating this is a check box and a value of unchecked.

[0110] B. "Check this then click save to copy the inactive config to the inactive" with a flag indicating this is a check box and a value of unchecked.

[0111] C. "Check then then click save to purge the inactive config" with a flag indicating this is a check box and a value of unchecked.

[0112] D. "Check this then click save to POR the adaptor" with a flag indicating this is a check box and a value of unchecked.

[0113] E. "Force adaptor offline" with a flag indicating this is a check box. This includes the currently assigned value per the specified Adaptor configuration data for this adaptor.

[0114] Continuing with non-illustrated, non-limiting examples of Administrative processing functionality:

[0115] Administration of Adaptor Configuration Data: Data Assignment of a /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Logical Units value:

[0116] Administrator

[0117] 1. Requests an assignment of any modifiable value under "/Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Logical Units" providing the new value along with the path to the data name.

[0118] Host DLL

[0119] 2. Assigns the specified data of the adaptor, indexed to the specified CU and Logical Unit, to the provided value.

[0120] 3. Saves the data to non-volatile storage through SA.

[0121] 4. If the change was to an emulated devices Name then, if that emulated devices is currently in use by a user per the emulated devices In-Use flag, use the emulated devices Client value to locate the client object then issue that client a message indicating the new emulated devices name.

[0122] 5. If the change was to an emulated devices Groups then, if that emulated devices is currently in use by a user per the emulated devices In-Use flag, use the emulated devices Client value to locate the client object and revalidate the client's authority exactly as is in accordance with Client Connection discussion. If the client no longer has the authority to access the device then send him a message to that effect and perform Client Disconnection processing.

[0123] Administration of Adaptor Configuration Data: Data Assignment of a /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Assignments value:

[0124] 1. Administrator requests an assignment of any modifiable value under "/Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Assignments", providing the new value along with the path to the data name.

[0125] Host DLL

[0126] 2. Ensures that every Logical Unit Base and Logical Unit Count is within the ranges established (and published) as acceptable to the adaptors and IBM mainframe computers. If not then reject the change

[0127] 3. Assigns the specified data of the adaptor, indexed to the specified CU, to the provided value.

[0128] 4. Saves the data to non-volatile storage through SA.

[0129] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Check this then click save to activate the inactive config

[0130] 1. Administrator requests to activate the inactive configuration of the specified adaptor.

[0131] Host DLL

[0132] 2. Uses the configuration datas for the specified adaptor.

[0133] 3. Indicates that the adaptor is unavailable for use by clients.

[0134] 4. For each client currently connected to an emulated devices on this adaptor:

[0135] A. Issue a message to the client indicating that they are being disconnected due to

administrator device management.

[0136] B. Perform Client Disconnection.

[0137] 5. Exchanges the contents of the active configuration with that of the inactive configuration.

[0138] 6. Saves the configurations to non-volatile storage through SA.

[0139] 7. Performs Adaptor Configuration Load.

[0140] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Check this then click save to copy the inactive config to the inactive

[0141] 1. Administrator requests to copy the active configuration to the inactive configuration of the specified adaptor.

[0142] Host DLL

[0143] 2. Uses the configuration datas for the specified adaptor.

[0144] 3. Copies the contents of the active configuration into the inactive configuration.

[0145] 4. Saves the inactive configuration to non-volatile storage through SA.

[0146] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Check then then click save to purge the inactive config

[0147] 1. Administrator requests to purge the inactive configuration of the specified adaptor.

[0148] Host DLL

[0149] 2. Uses the inactive configuration data for the specified adaptor.

[0150] 3. Clear it out to default values as does Start Server when a configuration doesn't exist. In summary, all of the CUTypes are assigned to 7412 and everything else is assigned to 0.

[0151] 4. Saves the inactive configuration to non-volatile storage through SA.

[0152] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Check this then click save to POR the adaptor

[0153] Administrator requests to perform a Power On Reset (POR, or an offline/online recycle) of the specified adaptor.

[0154] Host DLL performs Adaptor Configuration Load for the specified adaptor.

[0155] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Force adaptor offline

[0156] Administrator

[0157] 1. Requests a change to the flag that controls whether or not the adaptor is to be forced offline to the mainframe to which it is connected.

[0158] Host DLL

[0159] 2. Assigns the supplied setting to the data for the specified adaptor.

[0160] 3. Saves the value to non-volatile stored through SA.

[0161] 4. Performs Adaptor Configuration Load.

[0162] While this invention has been described to illustrative embodiments, this description is not to be construed in a limiting sense. Various modifications and combinations of the illustrative embodiments as well as other embodiments will be apparent to those skilled in the art upon referencing this disclosure. It is therefore intended that this disclosure encompass any such modifications or embodiments.

[0163] It will be apparent to those skilled in the art that various modifications and variations can be made in the construction, configuration, and/or operation of the present invention without departing from the scope or spirit of the invention. For example, in the embodiments mentioned above, variations in the materials used to make each element of the invention may vary without departing from the scope of the invention. Thus, it is intended that the present invention cover the modifications and variations of the invention provided they come within the scope of the appended claims and their equivalents.